به نام خدا

نویسنده : سحر فخریه کاشان

موضوع : python assistant

software documentation بخش مهمی از مهندسی نرم‌افزار است. مقوله‌های که معمولاً برای مستندسازی مورد توجه قرار گرفته می‌شوند عبارتند از:

نیازمندی‌ها –عباراتی که قابلیت‌ها، ویژگی‌ها یا کیفیت‌های یک سیستم را تعریف می‌کنند و زیر ساخت و پایه‌ای برای آنچه که باید اجرا گردد یا اجرا شده است، تلقی می‌گردند.

معماری/طرح – معماری نرم‌افزار و دورنمای آن که نمایش دهندهٔ نحوهٔ تعامل نرم‌افزار با محیط و ساختار و اجزاء نرم‌افزار و روابط میان این اجزاء است.

فنی – مستندسازی کد، الگوریتم‌ها، رابط‌های کاربری و رابط برنامه‌نویسی کاربردی.

کاربر نهایی – دستور راهنمایی برای کاربران هدف، راهبرهای سیستم و پشتیبان‌های فنی

نیازمندی ها :

برنامه ای که در این فایل ضمیمه شده، نمونه اولیه و مشابهی از google assistant است این برنامه قابلیت این را دارد که به سخنان شما گوش دهد و پس از پردازش آن به شما پاسخ دلخواهتان را بدهد

از جمله کارهایی که این برنامه قابلیت انجامشان را دارد : احوال پرسی با مخاطب، گفتن وضعیت آب و هوا ، سرچ در ویکیپدیا،یادداشت برداری سخنان کاربر، باز کردن ide و firefox و پخش آهنگ است.

این برنامه درصورتی که دستوری بگیرد که آن را متوجه نشود از کاربر عذرخواهی کرده و دستور را ذخیره میکند تا در نسخه های بعدی ، بتواند آن دستور را نیز انجام دهد.

معماری :

این برنامه با زبان پایتون نوشته شده و در کنسول اجرا میشود. اساس این برنامه تعامل با مخاطب است که به همین منظور برای ارتباط بهتر و راحت تر کاربر دو کتابخانه برای شنیدن سخنان کاربر و تبدیل آن به متن و برعکس کتابخانه دیگری برای تبدیل متن به گفتار در نظر گرفته شده است

محیط داخل کنسول نیز طوری در نظر گرفته شده که اگر کاربر نتواند سخنان را بشنود فقط با دیدن صفحه کنسول میتواند به راحتی با آن ارتباط برقرار کند و نیاز خود را برطرف نماید

فنی :

کتابخانه های مورد استفاده در این کد :

```
import speech_recognition as sr
import pyttsx3
import requests
```

```
import subprocess
import os
import wikipedia
```

در ادامه میبینیم که این کتاب خانه ها در چه جاهایی به کار رفته اند.

```
def saveFile(Fname, cmd):
    completeName = os.environ["HOMEPATH"] + "\\Desktop\\" + Fname + ".txt"
    f = open(completeName, "a+")
    f.write(cmd + '\r\n')
    f.close()
    print('saved')
    SpeakText('saved')
```

در این تابع متن مورد نظر و یا دستور ناشناخته کاربر در فایل های txt. متفاوت save میشود

```
def SpeakText(command):
    # Initialize the engine
    engine = pyttsx3.init()
    engine.say(command)
    engine.runAndWait()


def speechToText():
    # Initialize the recognizer
    r = sr.Recognizer()

    MyText = ''
    # Exception handling to handle
    # exceptions at the runtime
    while MyText == '' or MyText is None:

        try:

            # use the microphone as source for input.
            with sr.Microphone() as source2:
                # wait for a second to let the recognizer
                # adjust the energy threshold based on
                # the surrounding noise level
                r.adjust_for_ambient_noise(source2, duration=5)

                # listens for the user's input
                audio2 = r.listen(source2)

                # Using google to recognize audio
                MyText = r.recognize_google(audio2)
                MyText = MyText.lower()

                if MyText != '' or MyText is not None:
                    print(MyText)
                    SpeakText(MyText)
                    return MyText
```

```
        except sr.RequestError:
            print("Sorry, the service is down")
            SpeakText("Sorry, the service is down")
        except sr.UnknownValueError:
            print('sorry i didnt understand do you wanna add this
command?(y/n)')
            SpeakText('sorry i didnt understand do you wanna add this
command?(y/n)')
            inp = input()
            if inp == 'y':
                saveFile('command', MyText)
            else:
                print('say your command again')
                SpeakText('say your command again')
```

در این دو تابع به ترتیب متن به صوت و صوت به متن تبدیل میشود

که در تابع دوم اگر دستور ناشناسی را متوجه شود و کاربر هم آن دستور را تایید کند با استفاده از تابع saveFile که در قسمت قبل معرفی شد دستور در یک فایل txt ، save میشود

```
def greeting(txt):
    if 'hello' in txt:
        print('hi')
        SpeakText('hi')
    if 'how are you' in txt:
        print('im fine thank you.')
        SpeakText('im fine thank you.')
```

تابع برای سلام و احوال پرسی

```
def weather(city):
    # base URL
    BASE_URL = "https://api.openweathermap.org/data/2.5/weather?"
    # City Name
    CITY = city
    # API key
    API_KEY = 'aadda1b351cf4a2c78ad672630606ef8'
    # upadting the URL
    URL = BASE_URL + "q=" + CITY + "&appid=" + API_KEY
    # HTTP request
    response = requests.get(URL)
    # checking the status code of the request
    if response.status_code == 200:
        # getting data in the json format
        data = response.json()
        # getting the main dict block
        main = data['main']
        # getting temperature
        temperature = main['temp']
        # getting the humidity
        humidity = main['humidity']
        # getting the pressure
        pressure = main['pressure']
```

```
        # weather report
        report = data['weather']
        print(f"{CITY:-^30}")
        print(f"Temperature: {temperature} K")
        print(f"Humidity: {humidity}%")
        print(f"Pressure: {pressure} hpa")
        print(f"Weather Report: {report[0]['description']}")
    else:
        # showing the error message
        print("Error in the HTTP request")
        print(response.status_code)

    print('press any button to continue')
    SpeakText('press any button to continue')
    input()
```

در این تابع از API برای وضعیت آب و هوا استفاده شده است که این API از سایت api.openweathermap.org گرفته میشود

این تابع به عنوان ورودی نام شهر مورد نظر را میگیرد و در خروجی : دما، رطوبت، فشار هوا، گزارشی از هوا میدهد

```
def openProgram(cmd):
    if 'firefox' in cmd:
        subprocess.Popen('C:\\Program Files\\Mozilla Firefox\\firefox.exe')
    else:
        subprocess.Popen('C:\\Program Files\\JetBrains\\PyCharm
2018.3.5\\bin\\pycharm64.exe')

    print('press any button to continue')
    SpeakText('press any button to continue')
    input()
```

این تابع با استفاده از کتاب خانه subprocess و گرفتن آدرس دقیق برنامه ها میتواند آن ها را اجرا کند که در این جا بنا به خواسته پروژه IDE باز میشود (البته برای کار بیشتر firefox را نیز قرار دادیم)

```
def playMusic():
    os.startfile(os.path.join('01. Love Poem.mp3'))
    print('press any button to continue')
    SpeakText('press any button to continue')
    input()
```

این تابع با فایل آهنگی که در همان فایل کد ذخیره شده است را اجرا میکند

و سپس منتظر میماند تا کاربر برگردد و ادامه دهد

```
def wiki(txt):
    while True:
        try:
            url = wikipedia.page(wikipedia.suggest(txt)).url
            research = wikipedia.page(wikipedia.suggest(txt)).content
            print(url)
            print(research)
```

```
            print('press any button to continue')
            SpeakText('press any button to continue')
            input()
            break
        except Exception as e:
            print('your search have many results please search better :)')
            print('some options you can use:')
            SpeakText('your search have many results please search better
:)')
            SpeakText('some options you can use:')
            print(str(e))
            print('do you wanna search again?(y/n)')
            SpeakText('do you wanna search again?(y/n)')
            inp = input()
            if inp == 'y':
                print('what you wanna search about?')
                SpeakText('what you wanna search about?')
                txt = speechToText()
            else:
                break
```

این تابع با استفاده از کتابخانه wikipedia متلبی که میخواهید را سرچ میکند و به شما گزارش و لینکی از متلب را ارائه میدهد(در کنسول)

در نهایت این برنامه داخل یک حلقه بینهایت قرار گرفته تا هر وقت که کاربر برنامه را نبندد ویا با آن خداحافظی نکند در حال اجرا باقی میماند و بعد از اجرای هر دستور دستورات را دوباره تکرار میکند و منتظر تصمیم کاربر میشود

این برنامه پس از گرفتن دستور به دنبال کلید واژه های تعریف شده اش داخل متن میگردد و پس از آن متوجه میشود که چه عملی را باید انجام دهد و سپس تابع مورد نظر آن دستور را صدا میزند

کاربر نهایی :

این برنامه بسیار روان و راحت است طوری که هر کاربری به راحتی میتواند با آن ارتباط برقرار کند

گاهی به علت کند بودن اینترنت ممکن است این برنامه دارای سرعت عمل و گیرایی کمتری شود اما با کمی صبر به شما پاسخ درست و نهایی را میدهد

کاربر میتواند با گفتن کلید واژه هایی که خود جینی (نام برنامه) به آن ها معرفی میکند و یا کلید واژه های دیگری که در زیر لیستشان آمده دستور مورد نظر خود را بگوید و پس از آن فقط شاهد انجام شدن دستورش توسط جینی باشد.

کلید واژه ها:

1. hello – how are you – hello how are you
2. weather
3. open IDE/firefox
4. play music - play
5. take note - note

6. search on wikipedia - wikipedia

7. goodbye

پایان.